A. Experiment Details

In each experiment, the MSDNet backbone models are trained on the training data to minimise the L2 regularised sum of cross-entropy losses computed for all exits of the model. The L2 regularisation is implicit through weight decay in the SGD optimizer, and the loss for a single input sample is $\mathcal{L}_i = \sum_{k=1}^{n_{\text{block}}} -\ln p_k(\hat{\mathbf{y}}_i = \mathbf{y}_i \,|\, \mathbf{x}_i)$, where $p_k(\hat{\mathbf{y}}_i = \mathbf{y}_i \,|\, \mathbf{x}_i)$ is the predicted softmax confidence on the correct label $\mathbf{y}_i$ at classifier k. Moreover, validation data is used after every epoch to assess the performance of the model. Subsequently, we select the model that achieved the highest Top-1 accuracy on the validation set at the last classifier as the final model.

After training, we evaluate each model using the test set on a budgeted batch classification setup. For this, the computational budget is fixed, and each model must aim to classify the test set samples within the given budget. To achieve this, we use the validation set to calculate thresholds $t_k, k = 1, 2, \ldots, n_{\text{block}}$, one for each exit, such that the overall cost of classifying all test samples does not exceed the predefined budget B in expectation. To utilise the thresholds t_k, we need to employ a metric of uncertainty assigned to each prediction. If the uncertainty does not exceed the pre-calculated threshold at stage k we exit the model at the current stage and otherwise pass the test sample to the next block and continue with the computation. In our experiments, this uncertainty metric is the prediction confidence: the maximum softmax output value for each sample in the case of a vanilla MSDNet. For our model leveraging Laplace approximations we compute the maximum of the predictive posterior probabilities. As increasing prediction confidence corresponds to decreasing uncertainty, a sample is exited at exit k if the predicted confidence exceeds the confidence threshold t_k that has been calculated on the validation set.

A grid search is used to select values for the temperature scaling hyperparameter T_k and Laplace approximation prior variance σ_k. The values of T_k used in the grid search for all models are the following: [0.3, 0.5, 0.7, 1.0, 1.3, 1.5, 1.7, 2.0, 2.5, 3.0]. The values of Laplace prior variance σ_k used in the grid search are [0.5, 0.7, 1.0, 1.3, 1.5, 1.7, 2.0, 2.5, 3.0, 4.0]. If the temperature scale or Laplace prior variance are not optimised for in a grid search, as in many ablation study model options, they are set to their default values of 1.0 and 2.0 respectively. When MIE is not used, the grid search for T_k and σ_k is performed independently for each exit k minimising the NLPD on the validation set. When using MIE, T_k and σ_k are optimised one exit at a time sequentially, starting from the first exit, minimizing the ensemble prediction NLPD. When performing the grid search for exit j, the already optimised values of T_k and σ_k for exits $k = 1 \ldots j-1$ remain fixed, and the values T_j and σ_j are optimised by selecting the pair of values that minimises the MIE prediction NLPD for exit j.

To obtain the numbers in Tabs. 1, 2, 6 and 7, results for each model over a range of budgets are averaged. The budget range is different for ImageNet/Caltech-256 and CIFAR-100 models, and for each of the small, medium, and large size models. The computational budget ranges for averaging results for each model are listed in Tab. 3. All experiments are implemented with PyTorch [38].

A.1. CIFAR-100 Model and Training Details

The CIFAR-100 training set of 50,000 images is split into 45,000 training images and 5,000 validation images. The test set has 10,000 images. On CIFAR-100, models are trained for 300 epochs using a batch size of 64 images, and the learning rate is initially set to 0.1 and is decayed to one-tenth at epochs 150 and 225. The optimiser is SGD with a momentum of 0.9 and weight decay of 10^{-4}. The MSDNet backbone of all CIFAR-100 models has three scales of features. The number of channels after the first layer is 16, and the number of channels added by each layer is 6.

On CIFAR-100 the 'small', 'medium', and 'large' DNN backbones have 4, 6, and 8 blocks and classifiers each, respectively. The small architecture has a total of 10 layers, and transition layers that reduce the number of scales by one are performed at layers 5 and 9. The medium architecture has a total of 21 layers, and transition layers are performed at layers 8 and 15. The large architecture has a total of 36 layers, with transition layers performed at layers 13 and 25. Tab. 4 shows more detailed information separately for each block of the small, medium, and large models.

A.2. ImageNet Model and Training Details

The ImageNet training set of 1,281,167 images is split into 1,231,167 training images and 50,000 validation images. The test set is the standard test set of 50,000 images. In preprocessing, the sample images that are of varying resolutions are resized to a resolution of 224 by 224 pixels. On ImageNet, models are trained for 90 epochs using a batch size of 256, and the learning rate is initially set to 0.1 and it

Table 3. Ranges of computational budgets (FLOPs) over which results are averaged for different models to obtain the results shown in Tabs. 1, 2, 6 and 7.

CIFAR-100

Model size	Lower budget limit	Upper budget limit
Small	$7.0 \cdot 10^6$	$2.6 \cdot 10^7$
Medium	$2.5 \cdot 10^7$	$0.6 \cdot 10^8$
Large	$0.5 \cdot 10^8$	$1.4 \cdot 10^8$

ImageNet and Caltech-256

Model size	Lower budget limit	Upper budget limit
Small	$3.5 \cdot 10^8$	$1.1 \cdot 10^9$
Medium	$7.5 \cdot 10^8$	$2.2 \cdot 10^9$
Large	$1.5 \cdot 10^9$	$2.6 \cdot 10^9$

Table 4. Additional details on the different MSDNet backbone architectures used for different model sizes on CIFAR-100, ImageNet, and Caltech-256. Each row in the table shows details of a specific block in the architecture. L is the number of layers, FLOPs is the computational cost of processing one input sample, and n_{params} is the number of model parameters. FLOPs and n_{params} are cumulative numbers *i.e.* they include the numbers from the previous blocks. This means that the cost for the entire architecture is the cost shown for the last block.

CIFAR-100	Small			Medium			Large		
Block number	L	FLOPs (10^6)	n_{params} (10^6)	L	FLOPs (10^6)	n_{params} (10^6)	L	FLOPs (10^6)	n_{params} (10^6)
1	1	6.86	0.30	1	6.86	0.30	1	6.86	0.30
2	2	14.35	0.65	2	14.35	0.65	2	14.35	0.65
3	3	26.13	1.02	3	27.29	1.11	3	27.29	1.11
4	4	38.04	1.42	4	46.56	1.61	4	48.45	1.73
5	-	-	-	5	67.43	2.11	5	81.57	2.39
6	-	-	-	6	89.09	2.85	6	112.64	3.18
7	-	-	-	-	-	-	7	152.92	4.10
8	-	-	-	-	-	-	8	192.69	5.31
ImageNet	Small			Medium			Large		
Block number	L	FLOPs (10^6)	n_{params} (10^6)	L	FLOPs (10^6)	n_{params} (10^6)	L	FLOPs (10^6)	n_{params} (10^6)
1	4	339.90	4.24	6	514.66	7.08	7	615.6	8.76
2	4	685.46	8.77	6	1171.18	15.69	7	1436.39	20.15
3	4	1008.16	13.07	6	1844.52	24.01	7	2283.21	31.73
4	4	1254.47	16.75	6	2501.40	42.19	7	2967.42	41.86
5	4	1360.53	23.96	6	2742.06	56.53	7	3253.79	62.31
Caltech-256	Small			Medium			Large		
Block number	L	FLOPs (10^6)	n_{params} (10^6)	L	FLOPs (10^6)	n_{params} (10^6)	L	FLOPs (10^6)	n_{params} (10^6)
1	4	339.62	3.95	6	514.28	6.70	7	615.26	8.33
2	4	684.88	8.20	6	1170.40	14.90	7	1435.49	19.25
3	4	1007.32	12.24	6	1843.36	22.86	7	2281.85	30.37
4	4	1253.41	15.69	6	2499.59	40.38	7	2965.67	40.11
5	4	1359.05	22.48	6	2739.66	54.13	7	3251.32	59.84

is decayed to one-tenth at epochs 30 and 60. The optimiser is SGD with a momentum of 0.9 and weight decay of 10^{-4}. The MSDNet backbone of all ImageNet models has four scales of features and five blocks. The number of channels after the first layer is 32, and the number of channels added by each layer is 16.

The small, medium, and large models for ImageNet have five blocks and five classifiers each, but vary in the number of layers in each block (model architecture design follows [18]). The small architecture has a total of 20 layers, and transition layers that reduce the number of scales by one are performed at layers 6, 11, and 16. The medium architecture has a total of 30 layers, and transition layers are performed at layers 9, 17, and 25. The large architecture has a total of 35 layers, with transition layers at layers 10, 19, and 28. Tab. 4 shows more detailed information separately for each block of the small, medium, and large models.

A.3. Caltech-256 Model and Training Details

The Caltech-256 data set of 30,607 images is split to a training set of 23,107, a validation set of 2,500, and a test set of 5,000 samples. In preprocessing, the sample images that are of varying resolutions are resized to a resolution of 224 by 224 pixels, the same as for ImageNet data. The Caltech-256 data set has images from 257 categories.

On Caltech-256 the MSDNet backbone models are the same as for ImageNet, with the difference of the output dimensionality being 257 instead of 1,000, which affects the number of parameters and FLOPs slightly. Caltech-256 models are trained for 180 epochs with a batch size of 128. This difference in training compared to ImageNet models is due to the smaller number of training samples and the smaller number of classes. The learning rate is initially set to 0.1 and is decayed to one-tenth at epochs 90 and 135. Apart from the mentioned differences, the Caltech-256 models are trained the same as the ImageNet models.

A.4. Details on Baseline Models and Their Training

We use ResNet and DenseNet models as baseline architectures for CIFAR-100, ImageNet, and Caltech-256 experiments. Tab. 5 shows the model architecture details for all the used baseline models. Both ResNet and DenseNet models are implemented using the implementations from Torchvision [32]. All ResNet models are built using the basic residual layer with two consecutive 3 by 3 convolutions, except for the ImageNet/Caltech-256 ResNet50, which uses the bottleneck residual layer. For ImageNet and Caltech-256 DenseNet models, the growth rate is 32 and the number of initial features is 64. For CIFAR-100 DenseNet models, the growth rate is 12 and the number of initial features is 24. For ImageNet and Caltech-256, the Torchvision implementations are used as they are. On Caltech-256 the baseline models are the same as for ImageNet, with the difference of the output dimensionality being 257, which affects the number of parameters and FLOPs slightly.

For CIFAR-100, the architectures need some modifica-

Table 5. Details of the baseline ResNet and DenseNet models that are shown for comparison in Fig. 7. FLOPs is the computational cost of processing one input sample, and n_{params} is the number of model parameters. $L_{\text{B}i}$ is the number of layers in block i of the model (a block here refers to all layers between transition layers that change the feature map size). For CIFAR-100 models $i = 1 \dots 3$ and for ImageNet models $i = 1 \dots 4$.

CIFAR-100 baseline models

Model name	FLOPs	n_{params}	L_{B1}	L_{B2}	L_{B3}
ResNet8	$12.6 \cdot 10^6$	$8.39 \cdot 10^4$	1	1	1
ResNet14	$26.8 \cdot 10^6$	$18.1 \cdot 10^4$	2	2	2
ResNet20	$41.0 \cdot 10^6$	$27.8 \cdot 10^4$	3	3	3
ResNet26	$55.2 \cdot 10^6$	$37.6 \cdot 10^4$	4	4	4
ResNet38	$83.7 \cdot 10^6$	$57.0 \cdot 10^4$	6	6	6
ResNet50	$112 \cdot 10^6$	$76.4 \cdot 10^4$	8	8	8
ResNet62	$141 \cdot 10^6$	$95.9 \cdot 10^4$	10	10	10
ResNet86	$197 \cdot 10^6$	$135 \cdot 10^4$	14	14	14
ResNet110	$254 \cdot 10^6$	$174 \cdot 10^4$	18	18	18
DenseNet10	$10.0 \cdot 10^6$	$2.34 \cdot 10^4$	1	1	1
DenseNet16	$20.3 \cdot 10^6$	$4.88 \cdot 10^4$	2	2	2
DenseNet22	$31.7 \cdot 10^6$	$7.80 \cdot 10^4$	3	3	3
DenseNet28	$44.4 \cdot 10^6$	$11.1 \cdot 10^4$	4	4	4
DenseNet40	$73.4 \cdot 10^6$	$18.8 \cdot 10^4$	6	6	6
DenseNet52	$107 \cdot 10^6$	$28.0 \cdot 10^4$	8	8	8
DenseNet64	$146 \cdot 10^6$	$38.8 \cdot 10^4$	10	10	10
DenseNet76	$190 \cdot 10^6$	$51.0 \cdot 10^4$	12	12	12
DenseNet88	$239 \cdot 10^6$	$64.8 \cdot 10^4$	14	14	14

ImageNet baseline models

Model name	FLOPs	n_{params}	L_{B1}	L_{B2}	L_{B3}	L_{B4}
ResNet10	$8.93 \cdot 10^8$	$5.42 \cdot 10^6$	1	1	1	1
ResNet18	$18.2 \cdot 10^8$	$11.7 \cdot 10^6$	2	2	2	2
ResNet26	$27.4 \cdot 10^8$	$18.0 \cdot 10^6$	3	3	3	3
ResNet34	$36.7 \cdot 10^8$	$21.8 \cdot 10^6$	3	4	6	3
ResNet50	$41.0 \cdot 10^8$	$25.6 \cdot 10^6$	3	4	6	3
DenseNet57	$9.24 \cdot 10^8$	$2.44 \cdot 10^6$	2	6	10	8
DenseNet65	$13.8 \cdot 10^8$	$2.93 \cdot 10^6$	4	6	12	8
DenseNet81	$16.5 \cdot 10^8$	$4.18 \cdot 10^6$	4	8	16	10
DenseNet97	$25.3 \cdot 10^8$	$5.44 \cdot 10^6$	6	12	16	12
DenseNet121	$28.5 \cdot 10^8$	$7.98 \cdot 10^6$	6	12	24	16

Caltech-256 baseline models

Model name	FLOPs	n_{params}	L_{B1}	L_{B2}	L_{B3}	L_{B4}
ResNet10	$8.92 \cdot 10^8$	$5.04 \cdot 10^6$	1	1	1	1
ResNet18	$18.2 \cdot 10^8$	$11.3 \cdot 10^6$	2	2	2	2
ResNet26	$27.4 \cdot 10^8$	$17.6 \cdot 10^6$	3	3	3	3
ResNet34	$36.7 \cdot 10^8$	$21.4 \cdot 10^6$	3	4	6	3
ResNet50	$41.0 \cdot 10^8$	$24.0 \cdot 10^6$	3	4	6	3
DenseNet57	$9.24 \cdot 10^8$	$2.08 \cdot 10^6$	2	6	10	8
DenseNet65	$13.8 \cdot 10^8$	$2.55 \cdot 10^6$	4	6	12	8
DenseNet81	$16.5 \cdot 10^8$	$3.69 \cdot 10^6$	4	8	16	10
DenseNet97	$25.3 \cdot 10^8$	$4.87 \cdot 10^6$	6	12	16	12
DenseNet121	$28.5 \cdot 10^8$	$7.22 \cdot 10^6$	6	12	24	16

tions due to the smaller input dimensionality. For CIFAR-100 ResNets, the first convolutional layer is replaced by a 3 by 3 convolution with stride 1 and 16 output channels, and the max pooling layers are removed. For CIFAR-100 DenseNets, the first convolutional layer is replaced by a 3 by 3 convolution with stride 1, and the first batch normalization, max pooling, and ReLU operations are removed.

On CIFAR-100, ResNet and DenseNet models are trained for 300 epochs, and the learning rate is initially set to 0.1 and is decayed to one-tenth at epochs 150 and 225. On ImageNet, the ResNet and DenseNet models are trained for 90 epochs, and the learning rate is initially set to 0.1 and it is decayed to one-tenth at epochs 30 and 60. On Caltech-256 the baseline models are trained for 180 epochs with a batch size of 128. The learning rate is initially set to 0.1 and is decayed to one-tenth at epochs 90 and 135. For all data sets all ResNet and DenseNet models use the SGD optimiser with a momentum of 0.9 and weight decay of 10^{-4}.

A.5. Details on the Used Performance Metrics

For reporting model performances, we use the Top-1 and Top-5 accuracies, the negative log-predictive density (NLPD), and the expected calibration error (ECE). Top-1 accuracy is the standard accuracy metric, and is the percentage of test predictions for which the highest model predicted probability was on the correct class. Top-5 accuracy is the percentage of test predictions, for which the correct class is among the five classes that the model assigned the highest probability. Negative log-predictive density (NLPD) is defined as:

$$\text{NLPD} = -\sum_{i=1}^{n_{\text{test}}} \log p(\hat{\mathbf{y}}_i = \mathbf{y}_i \,|\, \mathbf{x}_i), \tag{3}$$

where $p(\hat{\mathbf{y}}_i = \mathbf{y}_i \,|\, \mathbf{x}_i)$ is the model predicted probability on the correct label $\mathbf{y}_i$. NLPD is a metric that captures both the quality of uncertainty estimates as well as the correctness of the predictions, most heavily penalizing overconfident incorrect predictions.

Expected calibration error (ECE) is defined as:

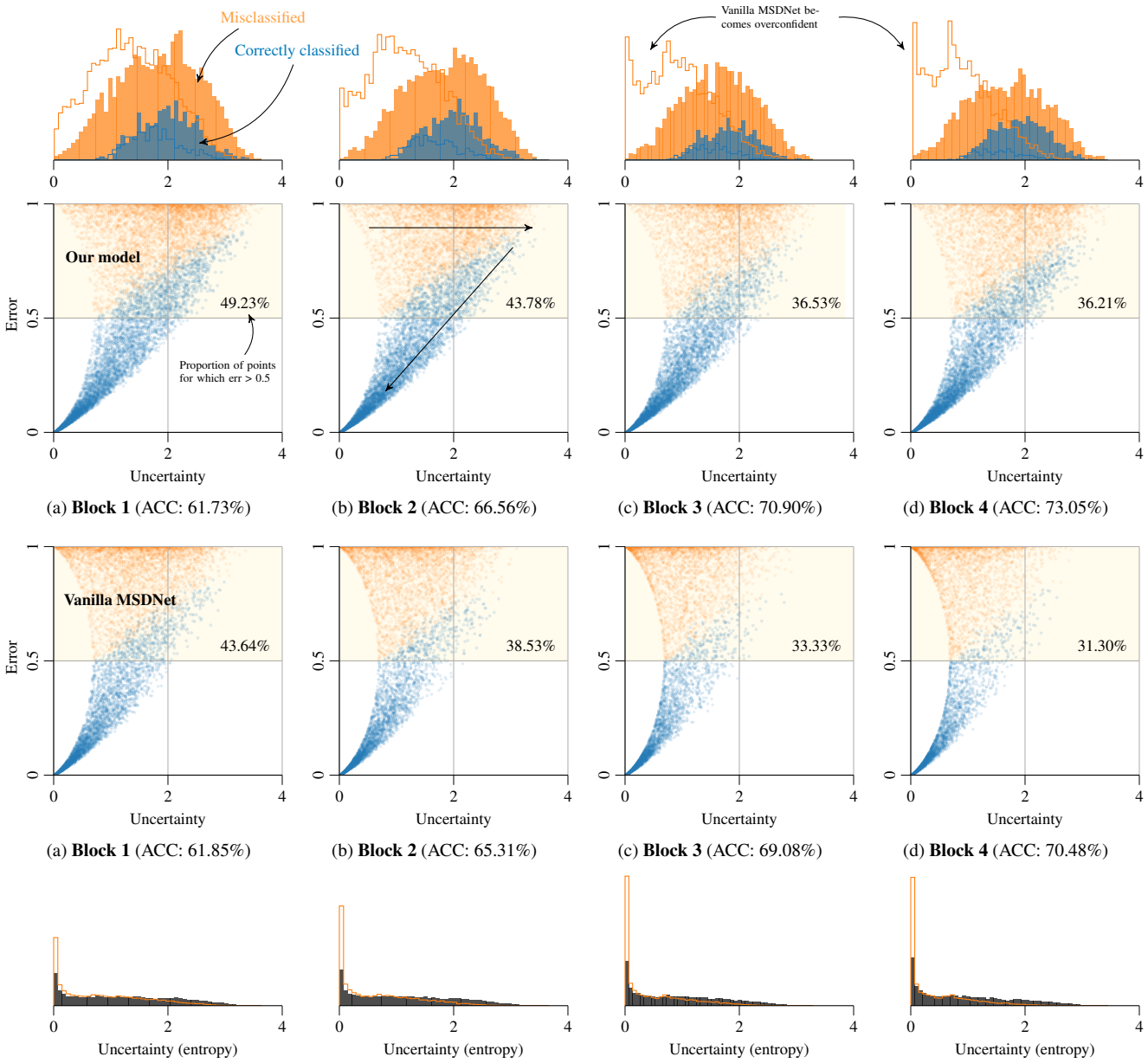
$$\text{ECE} = \sum_{j=1}^{m} b_j \|(p_j - \mu_j)\|, \tag{4}$$

where b_j is the fraction of test samples in bin $j = 1, \dots, m$, p_j is the Top-1 accuracy of the j^{th} bin, and μ_j is the average confidence of the predictions in the bin. We use $m = 10$ in our experiments. ECE assesses the calibration of each model, *i.e.*, how consistent the confidence scores are with the posterior probabilities.

B. Additional Results

Fig. 8 repeats the results from Fig. 5 and additionally shows corresponding scatter plots for the vanilla MSDNet for comparison. The bottom row in Fig. 8 shows predictive uncertainty histograms for all samples in the CIFAR-100 test set, comparing the vanilla MSDNet model to our model. The results are obtained using the small CIFAR-100 model. Comparing the scatter plots of our model with those of the vanilla model, we see that the vanilla model has more points in the top left corner of the plots, representing overconfident incorrect predictions. Looking at the bottom row histograms, we observe that the predictions from the vanilla MSDNet are overall more confident than those of our model.

Tab. 6 shows an extended version of the ablation study seen in Tab. 1, adding model versions that use Laplace approximation, but optimise only temperature scaling or



(a) **Block 1** (ACC: 61.73%) (b) **Block 2** (ACC: 66.56%) (c) **Block 3** (ACC: 70.90%) (d) **Block 4** (ACC: 73.05%)

(a) **Block 1** (ACC: 61.85%) (b) **Block 2** (ACC: 65.31%) (c) **Block 3** (ACC: 69.08%) (d) **Block 4** (ACC: 70.48%)

Figure 8. The second row scatter plots show correctly classified and misclassified test points for our model on an uncertainty vs. error axis (repeated from Fig. 8) and the third row shows the corresponding scatter plots for the vanilla MSDNet model. The uncertainty of the points in the top half of each scatter plot is summarized as a histogram in the top row (repeated from Fig. 8). Our model result is shown in the solid histograms, while the vanilla MSDNet results are shown as a histogram outline. Uncertainty histograms of all points in the uncertainty–error scatter plots are shown in the bottom row, comparing our model (black histogram) to the vanilla MSDNet model (orange outline). We can see that the vanilla MSDNet has overall much less uncertainty in its predictions

Laplace prior variance in a grid search, or optimise neither using fixed default values. We also include an ablation result where only temperature scaling is used on the vanilla model predictions. Tab. 6 also shows results for optimising the temperature scaling parameter when using MIE but without Laplace approximation. Similarly for Caltech-256, Tab. 7 shows a more extensive version of Tab. 2.

Looking at the results in these tables, we notice that although Laplace approximation alone often slightly decreases top-1 accuracy, when used together with MIE it increases top-1 accuracy above what MIE alone would achieve, suggesting that these two methods are suitable to be used together. In Tab. 6 the result using only temperature scaling for ImageNet has identical results with the vanilla model,

Table 6. Table of Top-1/Top-5 accuracy, negative log-predictive density (NLPD), and expected calibration error (ECE) for different models on CIFAR-100 and ImageNet data. All numbers are averages over a range of computational budgets in the budgeted batch classification setup. 'MIE Laplace T_{opt} σ_{opt}'-model corresponds to 'Our model' that is referred to in other figures. T_{opt} and σ_{opt} refer to grid search optimisation of the temperature scale and Laplace prior variance, respectively. n_{train} is the number of training samples, d is the input dimensionality, c is the number of classes, and n_{batch} is the batch size. The red and green numbers show the decrease or increase in performance compared to MSDNet (vanilla). The best performing model for each metric and each model size, on each dataset, is shown in bold.

	(n_{train}, d, c, n_{batch})	CIFAR-100 (50000, 3072, 100, 64)				IMAGENET (1281167, 150528, 1000, 256)			
		Top-1 ACC ↑	Top-5 ACC ↑	NLPD ↓	ECE ↓	Top-1 ACC ↑	Top-5 ACC ↑	NLPD ↓	ECE ↓
Small	MSDNet (vanilla)	69.25	90.48	1.498	0.182	68.15	88.22	1.338	0.019
	+ T_{opt}	69.06 −0.19	90.62 +0.14	1.207 −0.291	0.079 −0.103	68.15 +0.00	**88.22** +0.00	1.338 −0.000	0.019 −0.000
	+ Laplace	69.21 −0.04	90.46 −0.02	1.419 −0.079	0.155 −0.027	68.14 −0.01	88.21 −0.01	**1.335** −0.003	0.016 −0.003
	+ Laplace T_{opt}	69.02 −0.22	90.65 +0.17	1.196 −0.302	0.060 −0.121	68.13 −0.01	88.18 −0.04	1.337 −0.001	0.016 −0.004
	+ Laplace σ_{opt}	69.21 −0.04	90.42 −0.06	1.415 −0.082	0.154 −0.028	68.13 −0.02	88.17 −0.05	1.337 −0.001	0.016 −0.003
	+ Laplace T_{opt} σ_{opt}	69.06 −0.19	90.58 +0.10	1.208 −0.289	0.073 −0.109	68.10 −0.05	88.18 −0.04	1.337 −0.001	**0.015** −0.005
	+ MIE	69.97 +0.72	90.88 +0.40	1.218 −0.280	0.080 −0.102	68.27 +0.12	88.13 −0.10	1.355 +0.017	0.055 +0.036
	+ MIE T_{opt}	69.74 +0.50	**91.11** +0.63	**1.133** −0.365	0.028 −0.154	68.25 +0.10	88.04 −0.18	1.353 +0.015	0.038 +0.019
	+ MIE Laplace	**69.99** +0.74	90.88 +0.40	1.189 −0.308	0.056 −0.125	68.26 +0.11	88.11 −0.11	1.361 +0.023	0.070 +0.051
	+ MIE Laplace T_{opt}	69.83 +0.58	91.10 +0.62	1.135 −0.363	0.021 −0.161	68.22 +0.07	88.06 −0.16	1.357 +0.019	0.055 +0.036
	+ MIE Laplace σ_{opt}	69.89 +0.64	90.94 +0.46	1.192 −0.306	0.059 −0.122	68.25 +0.10	88.14 −0.08	1.360 +0.022	0.070 +0.051
	+ MIE Laplace T_{opt} σ_{opt}	69.84 +0.59	91.09 +0.61	1.133 −0.364	**0.017** −0.165	**68.31** +0.16	88.11 −0.11	1.356 +0.018	0.052 +0.032
Medium	MSDNet (vanilla)	74.12	91.94	1.549	0.190	72.78	91.01	1.123	0.033
	+ T_{opt}	73.96 −0.17	92.05 +0.10	1.063 −0.486	0.078 −0.112	72.78 −0.00	91.01 +0.00	1.123 +0.000	0.033 +0.000
	+ Laplace	73.96 −0.16	91.94 −0.00	1.436 −0.113	0.172 −0.018	72.69 −0.09	91.04 +0.03	**1.117** −0.006	**0.012** −0.021
	+ Laplace T_{opt}	73.98 −0.15	92.01 +0.07	1.056 −0.493	0.070 −0.120	72.68 −0.10	90.98 −0.03	1.117 −0.005	0.013 −0.020
	+ Laplace σ_{opt}	74.18 +0.05	91.85 −0.09	1.405 −0.144	0.164 −0.026	72.70 −0.08	91.00 −0.01	1.117 −0.005	0.013 −0.020
	+ Laplace T_{opt} σ_{opt}	73.92 −0.20	92.01 +0.06	1.070 −0.479	0.083 −0.107	72.72 −0.07	91.03 +0.03	1.118 −0.005	0.018 −0.015
	+ MIE	75.03 +0.91	92.97 +1.03	1.011 −0.538	0.050 −0.140	72.98 +0.20	91.12 +0.11	1.119 −0.004	0.042 +0.009
	+ MIE T_{opt}	74.94 +0.82	93.23 +1.29	**0.941** −0.608	0.028 −0.162	72.99 +0.21	91.09 +0.08	1.119 −0.004	0.038 +0.005
	+ MIE Laplace	74.99 +0.86	93.01 +1.07	0.990 −0.559	0.032 −0.158	72.95 +0.17	**91.15** +0.14	1.128 +0.005	0.065 +0.032
	+ MIE Laplace T_{opt}	74.96 +0.84	93.19 +1.24	0.947 −0.602	**0.015** −0.175	72.88 +0.10	91.06 +0.06	1.124 +0.001	0.045 +0.012
	+ MIE Laplace σ_{opt}	**75.04** +0.92	92.95 +1.01	0.989 −0.560	0.031 −0.159	72.97 +0.19	91.12 +0.11	1.126 +0.003	0.064 +0.030
	+ MIE Laplace T_{opt} σ_{opt}	74.99 +0.86	**93.23** +1.29	0.944 −0.605	0.026 −0.164	**73.04** +0.26	90.96 −0.05	1.121 −0.002	0.031 −0.003
Large	MSDNet (vanilla)	75.36	92.78	1.475	0.178	74.33	91.57	1.066	0.050
	+ T_{opt}	75.27 −0.10	92.76 −0.02	0.984 −0.491	0.059 −0.119	74.33 +0.00	91.57 +0.00	1.066 +0.000	0.050 +0.000
	+ Laplace	75.41 +0.05	92.76 −0.02	1.347 −0.128	0.157 −0.021	74.25 −0.08	91.55 −0.02	1.052 −0.014	0.019 −0.031
	+ Laplace T_{opt}	75.28 −0.08	92.79 +0.01	0.999 −0.476	0.077 −0.101	74.25 −0.08	91.55 −0.02	1.053 −0.012	0.019 −0.031
	+ Laplace σ_{opt}	75.36 −0.01	92.75 −0.04	1.338 −0.137	0.157 −0.020	74.25 −0.08	91.55 −0.03	1.053 −0.013	**0.017** −0.033
	+ Laplace T_{opt} σ_{opt}	75.32 −0.05	92.83 +0.05	0.996 −0.479	0.075 −0.103	74.29 −0.04	91.53 −0.04	1.053 −0.013	0.020 −0.030
	+ MIE	76.32 +0.95	93.50 +0.72	0.949 −0.525	0.061 −0.117	74.82 +0.49	91.88 +0.30	**1.029** −0.037	0.028 −0.022
	+ MIE T_{opt}	76.22 +0.85	93.75 +0.97	0.886 −0.589	0.032 −0.145	**74.90** +0.58	91.87 +0.30	1.029 −0.037	0.022 −0.028
	+ MIE Laplace	**76.43** +1.07	93.55 +0.76	0.924 −0.551	0.040 −0.137	74.76 +0.43	**91.90** +0.33	1.035 −0.030	0.052 +0.002
	+ MIE Laplace T_{opt}	76.30 +0.93	93.74 +0.96	0.887 −0.588	0.036 −0.142	74.86 +0.53	91.78 +0.21	1.032 −0.034	0.026 −0.024
	+ MIE Laplace σ_{opt}	76.33 +0.96	93.54 +0.75	0.925 −0.550	0.043 −0.135	74.81 +0.49	91.87 +0.29	1.033 −0.032	0.050 −0.000
	+ MIE Laplace T_{opt} σ_{opt}	76.34 +0.98	**93.84** +1.05	**0.885** −0.590	**0.025** −0.152	74.80 +0.47	91.81 +0.24	1.032 −0.034	0.032 −0.019

as the best temperature after optimization ended up being the default temperature, suggesting that the vanilla MSDNet on ImageNet is already quite well calibrated. This is reflected in the results in Tab. 6 also through the fact that our methods that attempt to improve decision-making through improved calibration, do not achieve major improvement in top-1 accuracy on ImageNet, as there is not much room to improve calibration over the vanilla MSDNet. This is likely explained by the vanilla MSDNet underfitting the ImageNet data, as even the largest MSDNet architecture we used for ImageNet is several magnitudes smaller than the state-of-the art models. On CIFAR-100 and Caltech-256 the MSDNet models are large enough to overfit, as is usually the case for most models on most datasets, and we see considerable improvements in also top-1 accuracy.

In order to investigate the contribution of better decision-making on the improvements in the predictive performance, we performed an experiment trying to separate the improvement due to better decision-making from the improvement due to better predictions at each individual intermediate exit of the dynamic neural network. In this experiment, we replaced the vanilla model decision-making with the decision-making of our model, while using the vanilla model predictions for calculating the results. Full results for CIFAR-100 are in Fig. 9 showing that our approach improves both decision-making (orange curve vs. light blue curve) and prediction quality (light blue curve vs. black curve). Interestingly, apart from improving accuracy, better decision-making also improves calibration and uncertainty estimates, as seen from the improved ECE and NLPD. However, this experiment is problematic in providing information on decision-making quality, as one model is making decisions using predictions from another model, potentially resulting in false interpretation of bad decision-making, if different models predict different samples correctly.

In addition to the experiments shown, we experimented using predictive variance or entropy as the uncertainty metric to make decisions on when to exit the MSDNet pipeline. In our experiments these metrics performed worse than the model predicted confidence, and hence we use model pre-

Table 7. Table of Top-1/Top-5 accuracy, NLPD, ECE for different models on Caltech-256. All numbers are averages over a range of computational budgets. 'MIE Lap T_{opt} σ_{opt}'-model corresponds to 'Our model'. T_{opt} and σ_{opt} refer to grid search optimisation of the temperature scale and Laplace prior variance, respectively. n_{train} is the number of training samples, d is the input dimensionality, c is the number of classes, and n_{batch} is the batch size. The red and green numbers show the decrease or increase in performance compared to MSDNet (vanilla). The best performing model for each metric and each model size, on each dataset, is shown in bold.

	(n_{train}, d, c, n_{batch})	CALTECH-256 (25607, 150528, 257, 128)			
		Top-1 ACC ↑	Top-5 ACC ↑	NLPD ↓	ECE ↓
Small	MSDNet (vanilla)	61.0	78.2	2.16	0.18
	+ T_{opt}	60.8 −0.3	78.4 +0.2	1.84 −0.31	**0.01** −0.16
	+ Lap	60.7 −0.3	78.0 −0.2	1.98 −0.18	0.06 −0.12
	+ Lap T_{opt}	60.5 −0.6	78.3 +0.0	1.86 −0.29	0.05 −0.13
	+ Lap σ_{opt}	60.6 −0.4	78.0 −0.2	1.99 −0.17	0.06 −0.12
	+ Lap T_{opt} σ_{opt}	60.5 −0.5	78.1 −0.1	1.86 −0.29	0.05 −0.13
	+ MIE	**61.9** +0.9	78.8 +0.6	1.94 −0.21	0.08 −0.10
	+ MIE T_{opt}	61.5 +0.5	**79.1** +0.9	**1.79** −0.37	0.04 −0.13
	+ MIE Lap	61.8 +0.7	78.8 +0.5	1.86 −0.30	0.04 −0.14
	+ MIE Lap T_{opt}	61.5 +0.5	79.1 +0.9	1.82 −0.34	0.08 −0.10
	+ MIE Lap σ_{opt}	61.8 +0.8	79.0 +0.8	1.86 −0.30	0.03 −0.14
	+ MIE Lap T_{opt} σ_{opt}	61.7 +0.6	79.0 +0.8	1.81 −0.34	0.09 −0.09
Medium	MSDNet (vanilla)	63.8	80.2	1.98	0.17
	+ T_{opt}	63.5 −0.3	80.2 +0.0	1.70 −0.28	**0.02** −0.15
	+ Lap	63.4 −0.4	79.4 −0.8	1.80 −0.18	0.03 −0.14
	+ Lap T_{opt}	63.3 −0.5	79.8 −0.3	1.73 −0.25	0.08 −0.09
	+ Lap σ_{opt}	63.4 −0.4	79.9 −0.3	1.79 −0.19	0.04 −0.13
	+ Lap T_{opt} σ_{opt}	63.4 −0.4	79.9 −0.3	1.74 −0.24	0.07 −0.10
	+ MIE	**65.1** +1.3	81.4 +1.2	1.72 −0.26	0.08 −0.09
	+ MIE T_{opt}	64.6 +0.9	**81.7** +1.5	**1.61** −0.37	0.03 −0.14
	+ MIE Lap	64.9 +1.2	81.2 +1.1	1.67 −0.31	0.06 −0.11
	+ MIE Lap T_{opt}	64.5 +0.7	81.3 +1.1	1.67 −0.31	0.09 −0.08
	+ MIE Lap σ_{opt}	64.7 +0.9	81.2 +1.0	1.67 −0.31	0.04 −0.13
	+ MIE Lap T_{opt} σ_{opt}	64.3 +0.5	81.3 +1.1	1.65 −0.33	0.08 −0.09
Large	MSDNet (vanilla)	64.9	80.7	1.90	0.16
	+ T_{opt}	64.8 −0.1	80.7 −0.0	1.64 −0.26	**0.03** −0.13
	+ Lap	64.3 −0.6	80.1 −0.6	1.72 −0.18	0.04 −0.12
	+ Lap T_{opt}	64.4 −0.5	80.0 −0.7	1.68 −0.22	0.07 −0.09
	+ Lap σ_{opt}	64.8 −0.1	80.1 −0.6	1.73 −0.17	0.03 −0.13
	+ Lap T_{opt} σ_{opt}	64.7 −0.2	80.7 +0.0	1.65 −0.25	0.04 −0.12
	+ MIE	65.9 +0.9	82.4 +1.8	1.62 −0.28	0.06 −0.10
	+ MIE T_{opt}	**65.9** +1.0	**82.5** +1.9	**1.54** −0.36	0.04 −0.12
	+ MIE Lap	65.9 +1.0	82.1 +1.5	1.59 −0.31	0.08 −0.08
	+ MIE Lap T_{opt}	65.9 +0.9	82.4 +1.8	1.59 −0.31	0.11 −0.05
	+ MIE Lap σ_{opt}	65.9 +0.9	82.4 +1.7	1.58 −0.31	0.07 −0.09
	+ MIE Lap T_{opt} σ_{opt}	65.6 +0.7	82.5 +1.8	1.58 −0.32	0.09 −0.07

dictive confidence for decision-making in all experiments shown in this paper.

C. Analysis on Laplace Approximation and MIE Computational Cost

Using the Laplace approximation requires calculating the approximate inverse Hessians $\mathbf{H}^{-1}$ and $\mathbf{\Sigma}_{\mathbf{b}}$ for the last layer weights and biases respectively, for each intermediate classifier. As this can be precomputed before observing the test data, the only additional test time cost of the Laplace approximation comes from transforming the Laplace approximated distribution for the last layer weights $\mathrm{N}(\boldsymbol{\theta} \mid \boldsymbol{\theta}_{\text{MAP}}, \mathbf{\Sigma})$ to an output predictive distribution $p(\hat{\mathbf{z}}_i \mid \mathbf{x}_i)$ and sampling from this distribution for n_{MC} times. Recall that $\hat{\mathbf{y}}_i = \mathrm{softmax}(\hat{\mathbf{z}}_i)$, where $\hat{\mathbf{z}}_i = \mathbf{W}\boldsymbol{\phi}_i + \mathbf{b}$.

Naïve approach We use the KFAC approximation to the inverse Hessian $\mathbf{H}^{-1} = \mathbf{V}^{-1} \otimes \mathbf{U}^{-1}$. Let $\mathbf{W} \in \mathbb{R}^{p \times c}$, $\mathbf{b} \in \mathbb{R}^{1 \times c}$ denote the weight matrix and bias terms of the k^{th} exit, and $\boldsymbol{\phi}_i \in \mathbb{R}^p$ denote the features of input sample $\mathbf{x}_i$ before the last linear layer of the k^{th} exit. Then the additional cost associated to a naïve implementation of the Laplace approximation at the k^{th} exit is based on

$$\hat{\mathbf{z}}_i \sim \mathrm{N}(\underbrace{\mathbf{W}_{\text{MAP}}^\top \boldsymbol{\phi}_i + \mathbf{b}_{\text{MAP}}}_{\text{also needed for vanilla}}, \underbrace{(\boldsymbol{\phi}_i^\top \mathbf{V} \boldsymbol{\phi}_i)\mathbf{U} + \mathbf{\Sigma}_{\mathbf{b}}}_{(p+1)(2p-1)+2c^2}) \quad (5)$$

with additional costs to compute the Cholesky factorisation ($\frac{1}{3}c^3$) and rescaling and shifting the samples drawn from a standard normal, resulting in a total of

$$\text{FLOPs}_{\text{naïve}} = 2c^2(n_{\text{MC}} + 1) + \frac{1}{3}c^3 + 2p^2 + p - 1. \quad (6)$$

additional FLOPs. The calculation of the mean in Eq. (5) does not add computation as this operation is also performed to obtain the vanilla MSDNet prediction. The cubic scaling of FLOPs with the number of classes is what makes the naïve approach prohibitively expensive to be viable in the budget restricted regime if the number of classes is large. The Cholesky factorization of the covariance matrix that is required for sampling can not be pre-computed before test time as the covariance matrix $(\boldsymbol{\phi}_i^\top \mathbf{V} \boldsymbol{\phi}_i)\mathbf{U} + \mathbf{\Sigma}_{\mathbf{b}}$ depends on the test samples.

Efficient approach Sampling from the Laplace predictive distribution can be made more efficient by absorbing the bias terms into the weight matrix, *i.e.*, $\hat{\mathbf{W}} \in \mathbb{R}^{p+1 \times c}$. This appends an additional dimension to $\boldsymbol{\phi}$ and $\mathbf{V}$ which we now denote by $\hat{\boldsymbol{\phi}} = (\boldsymbol{\phi}^\top, 1)^\top$ and $\hat{\mathbf{V}}$, respectively. Now, the output predictive distribution takes the form $\hat{\mathbf{z}}_i \sim \mathrm{N}(\hat{\mathbf{W}}_{\text{MAP}}^\top \hat{\boldsymbol{\phi}}_i, (\hat{\boldsymbol{\phi}}_i^\top \hat{\mathbf{V}} \hat{\boldsymbol{\phi}}_i)\mathbf{U})$, which means that the costly operations can all be pre-computed offline. Most notably, although the covariance matrix $(\hat{\boldsymbol{\phi}}_i^\top \hat{\mathbf{V}} \hat{\boldsymbol{\phi}}_i)\mathbf{U}$ still depends on the test samples, the test sample dependent term $\hat{\boldsymbol{\phi}}_i^\top \hat{\mathbf{V}} \hat{\boldsymbol{\phi}}_i$ is a scalar and can be taken out of the costly Cholesky factorization, which allows performing this on the matrix $\mathbf{U}$ defore test time. This means that for one MC sample l, the pre-softmax output can be evaluated as

$$\hat{\mathbf{z}}_i^{(l)} = \underbrace{\hat{\mathbf{W}}_{\text{MAP}}^\top \hat{\boldsymbol{\phi}}_i}_{\text{also needed for vanilla}} + \underbrace{\sqrt{\hat{\boldsymbol{\phi}}_i^\top \hat{\mathbf{V}} \hat{\boldsymbol{\phi}}_i}}_{(p+2)(2p+1)} \underbrace{(\mathbf{L}\mathbf{g}^{(l)})}_{2c^2-c}, \quad (7)$$

where $\mathbf{g}^{(l)} \sim \mathrm{N}(\mathbf{0}, \mathbf{I})$ and $\mathbf{L}$ is the pre-calculated Cholesky factor of $\mathbf{U}$. Even the samples $\mathbf{g}^{(l)}$ can be pre-drawn and pre-multiplied with $\mathbf{L}$, which further reduces the computational cost to

$$\text{FLOPs}_{\text{efficient}} = 2cn_{\text{MC}} + 2p^2 + 5p + 2. \quad (8)$$

Based on this analysis, Fig. 10 shows the incremental test time computational cost of applying Laplace approximation on top of the vanilla MSDNet model for CIFAR-100,

Table 8. Table of Top-1/Top-5 accuracy, negative log-predictive density (NLPD), and expected calibration error (ECE) for different models on CIFAR-100. 'Our model' corresponds to 'MIE Laplace T_{opt} σ_{opt}'-model in other result tables. These results show a decision-making experiment, where vanilla MSDNet and 'Our model' results are compared to results obtained by using a setup where our model is used for decision-making, and predictions are from vanilla MSDNet. The best-performing model for each metric and each model size is shown in bold.

	$(n_{\text{train}}, d, c, n_{\text{batch}})$	CIFAR-100 (50000, 3072,100, 64)			
		Top-1 ACC ↑	Top-5 ACC ↑	NLPD ↓	ECE ↓
Small	MSDNet (vanilla)	69.25	90.48	1.498	0.182
	Vanilla predictions, our model decisions	69.33 +0.09	90.60 +0.12	1.300 −0.197	0.108 −0.074
	Our model	**69.84** +0.59	**91.09** +0.61	**1.133** −0.364	**0.017** −0.165
Medium	MSDNet (vanilla)	74.12	91.94	1.549	0.190
	Vanilla predictions, our model decisions	74.51 +0.39	92.20 +0.25	1.460 −0.089	0.168 −0.022
	Our model	**74.99** +0.86	**93.23** +1.29	**0.944** −0.605	**0.026** −0.164
Large	MSDNet (vanilla)	75.36	92.78	1.475	0.178
	Vanilla predictions, our model decisions	75.72 +0.36	92.71 −0.08	1.388 −0.086	0.162 −0.015
	Our model	**76.34** +0.98	**93.84** +1.05	**0.885** −0.590	**0.025** −0.152

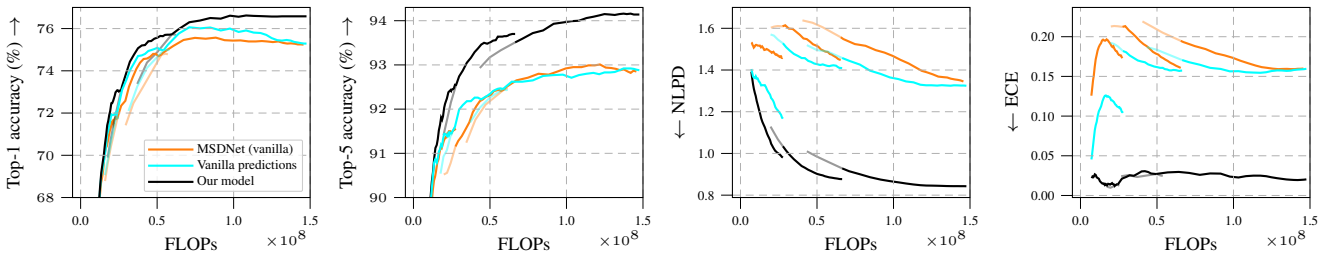
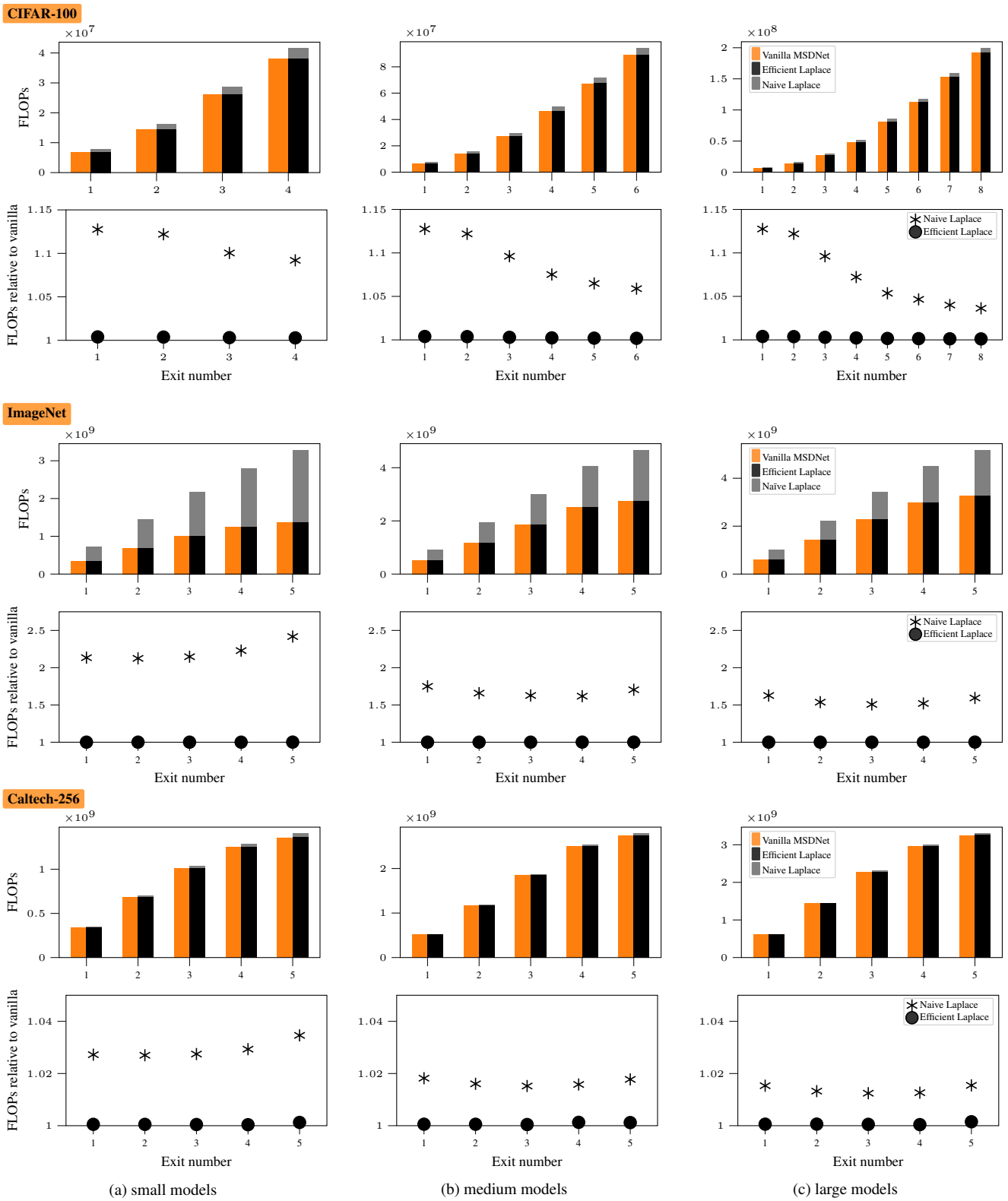


Figure 9. Accuracy (Top-1 & Top-5) and uncertainty metrics (NLPD and ECE) on a budgeted batch classification task as a function of average computational budget per image (FLOPs) on the CIFAR-100 data set with a small/medium/large model. These results show a decision-making experiment, where vanilla MSDNet and 'Our model' results are compared to results obtained by using a setup where our model is used for decision-making, and predictions are from vanilla MSDNet (labelled 'Vanilla predictions').

ImageNet, and Caltech-256, showing the increase in computation both for the efficient sampling approach and for the naïve sampling approach. For all data sets the small, medium, and large models are analysed separately, and for each model, the increase in computational cost at each intermediate exit is shown. The calculation of FLOPs in these figures differs from what is shown in the formulas above, as the FLOPs in the figure results are 'practical FLOPs' that take into account the ability of most hardware to calculate sequential multiplication and addition in a single operation, resulting in one FLOP instead of two for such a pair of operations. Practical FLOPs are used also for the FLOPs calculation of all other numerical results in this paper. The results of the figure are obtained considering that 50 MC samples are drawn from the Laplace approximated predictive output distribution. The figures show that naïve approach of Laplace approximation adds considerable computation, but this can be mitigated by using the efficient sampling approach. The remaining added computational cost of efficient Laplace that is hard to see in Fig. 10 is 0.1–0.4% on CIFAR-100, 0.06–0.16% on ImageNet, and 0.04–0.15% on Caltech-256 depending on the exit used. As a comparison, using a last layer MC dropout with 50 samples would add 3–9% computation on CIFAR-100, 4–10% on ImageNet, and 1–3% on Caltech-256.

Using MIE adds even less computation compared to our efficient Laplace. At each exit apart from the first one, the c dimensional output is multiplied by the weight averaging weight w_k, added to the cumulative total sum, which is then divided by the total weight $\sum_{j=1}^{k} w_j$, adding up to $3c$ additional FLOPs at each exit after the first one. This results to additional 0.001–0.002% computation on CIFAR-100, 0.0002–0.0009% on ImageNet, and 0.0001–0.0002% on Caltech-256.



(a) small models (b) medium models (c) large models

Figure 10. Analysis on the test time computational cost of the Laplace approximation for different models. First rows show bar graphs comparing the computational costs of vanilla MSDNet (orange) and MSDNet with Laplace approximation (black) at each intermediate exit. The solid black bars represent the computational cost of efficient sampling from the predictive distribution, while the grey bars show the cost of naïve sampling. The second rows show the relative computational cost of using Laplace approximation on top of the vanilla MSDNet, both for the efficient and the naïve sampling approach. The figures assume 50 MC samples are drawn from the predictive distribution.